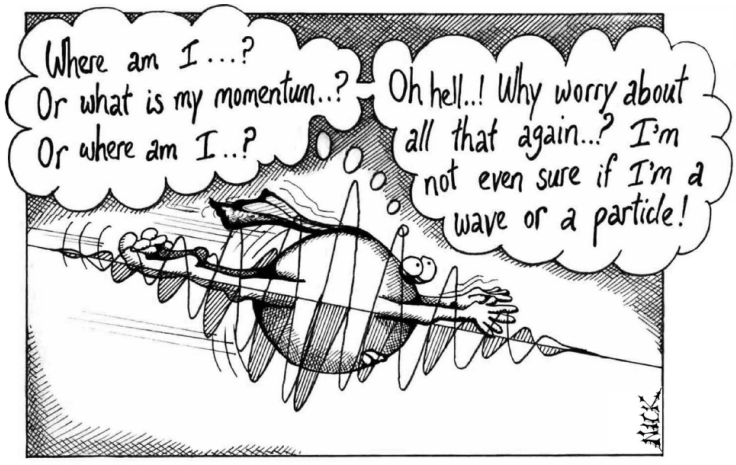



Chapter 12

Lecture 12 - Final lecture


Quantum effects...



Photon self-identity problems.

Tut 9: Q1 - bad code

The code is supposed to clear the password buffer:

```
// ... Zero out all the contents of the buffer:
bzero( buffer, MaxPasswordSize );
// ... Immediately return the memory to the OS:
free( buffer );
```

But ... the compiler optimizes the `bzero()` away - It reasons: If you are going to `free()` the memory, then *any previous unused assignments can be discarded*.

Leaves the password in memory.

Tut 9: Q2 - Instructor/submissions

Answer: *A solution might be for the instructor to send the students (securely) a secret key, which only they could know. The students might then use this key to encrypt their submission.*

If the encryption was symmetric, a student on the course could forge someone else's submission. So better to use asymmetric.

It might be useful to consider other variations. For example - what if the bad-guys could listen to the *"single secure message"*? What if the bad-guys had a collaborator? and so on...

Tut 9: Q3 - Describe and contrast

Example: AES is a block cipher similar to Rijndael, with a variable key length. Keys can have a length of 128, 192, or 256 bits to encrypt blocks with length of 128 bits. Adopted as a US standard, and approved by NSA for use (the only public protocol so approved).

Still fast, but considered significantly more difficult to attack than (say) DES, due to large key size and defined mathematical properties.

Quantum physics

Relevant for two reasons: Quantum *computing* and quantum *cryptography*.

1. Quantum *computers* may be able to compute HARD problems quickly (such as factorizing large composites).

How? The underlying data elements are quantum bits (qubits), not limited to just 0,1 states - instead considered to be a superposition of states. An operation performed on a qubit is performed on all the states simultaneously.

It is likely that no effective quantum computer has yet been built.

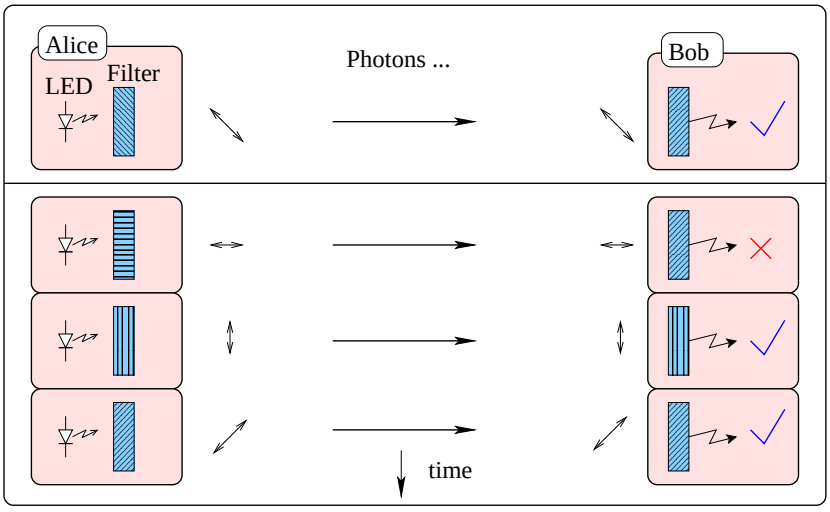
Quantum cryptography

2. Quantum *cryptography* uses laws of quantum mechanics - Heisenberg Uncertainty applies to some pairs of (atomic) particles. Measuring one property affects another.

A snooper is easily detected, and there are various protocols for using quantum effects to share keys.

Alice randomly chooses one of four polarizations: 0, 90, or 45, 135 degrees.

Alice transmitting 10000 photons



Bob and Harry the hacker

1. Alice records what she has done. Bob randomly chooses polarizations, and reads the results.

2. Bob tells Alice that he has used diag, diag, rectilinear, diag. Alice replies by telling Bob which ones were correct.

3. They now have 5000 (approx) bits in common.

Harry the hacker

1. If Harry the hacker senses (some of) the photons, he must choose which polarization to use, and will affect the photon.

2. Bob and Alice compare a subset of the bits that they think they know to detect snooping.

3. If no snooping, then rest of bits are OK.

Quantum cryptography systems are now commercially available, operating over reasonably long (40km) fibre.

More insecurity

...Who are you and how did you get in here?

...I'm a locksmith. And, I'm a locksmith. [Leslie Nielsen]

Design principles

http://web.mit.edu/Saltzer/www/publications/protection/index.html

Paper by Saltzer and Schroeder, summarized below:

- **Economy of mechanism:** Keep design as simple and small as possible. (audit code and protocols)
- **Fail-safe defaults:** Base access decisions on permission rather than exclusion. The default is no access. (MS port security)
- **Complete mediation:** Every access to every object must be checked for authority. (DNS cache poisoning)

Design principles

- **Open design:** The design should not be secret. (DVDs, Microsoft SAM hashes...)
- **Separation of privilege:** Two keys are better than one. No single event can compromise the system. (su - password and *wheel* group)
- **Least privilege:** Every program and every user of the system should operate using the least set of privileges necessary to complete the job. (Military need-to-know)

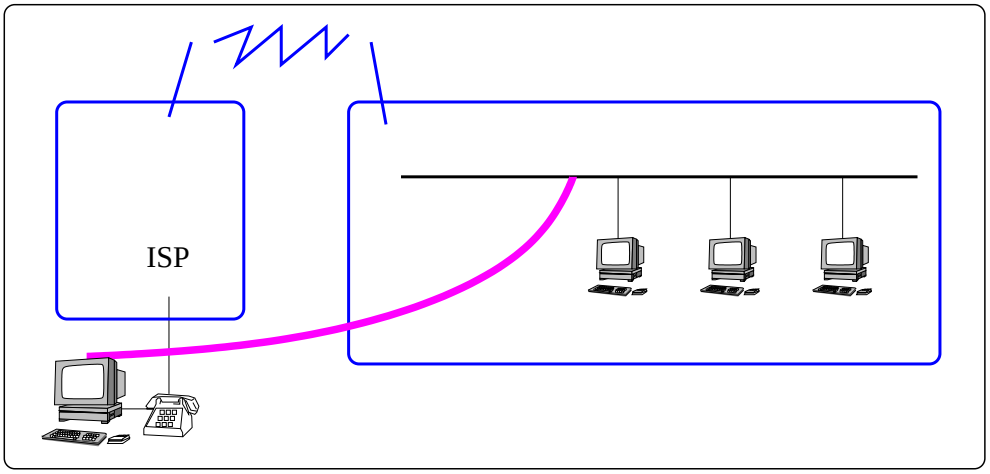
Design principles

- **Least common mechanism:** Minimize the amount of mechanism common to more than one user and depended on by all users. (online store and D.O.S.).
- **Psychological acceptability:** Human interface easy to use.

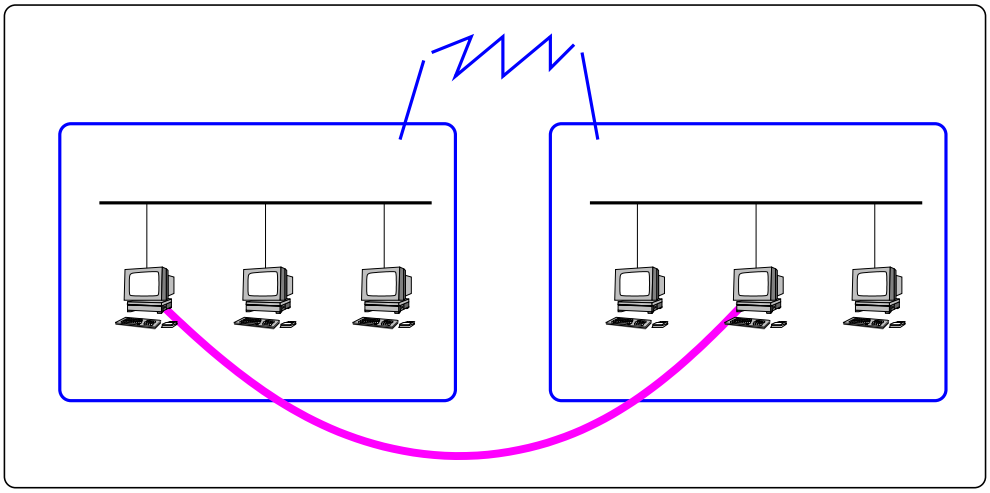
IPSec

- IPSec is a **set of standards** intended to support communication security between networked computers, particularly in the newer IPv6 (IP Next-Generation) network.
- IPSec software is available in Windows2000, Linux, and on routers on the Internet.
- **http://www.faqs.org/rfcs/rfc2401.html**
- IPSec may be used in a range of ways.

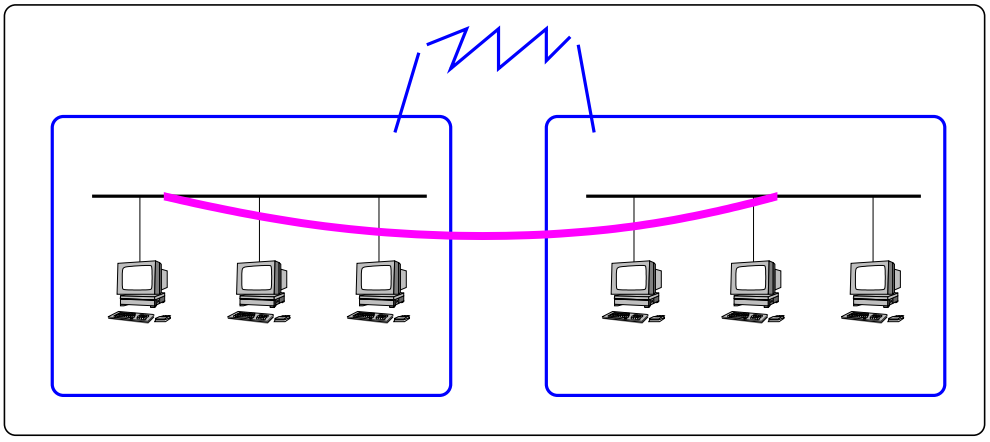
IPSec VPN



IPSec point-to-point



IPSec network-to-network



IPSec headers

There are two types of header, one used for **authentication**, and the other used for **encryption**:

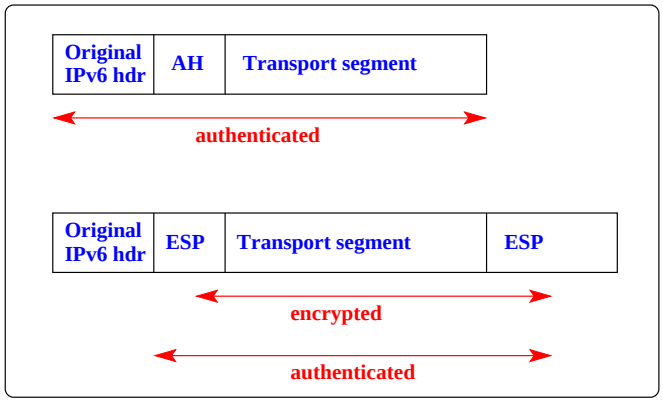
1. **AH** - the **Authentication Header** for data integrity, anti-replay and authentication

2. **ESP** - the **Encapsulating Security Payload** header, for confidentiality. ESP can also provide AH services.

Communicating parties agree on a **Security Association** (SA), one SA for each direction, and one SA for each type of communication.

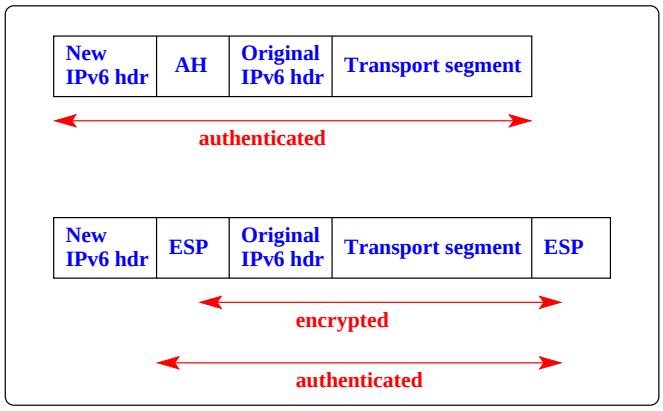
Modes of operation

- An end-to-end SA - Transport mode



Modes of operation

- An SA between security gateways - Tunnel mode



SAs form a kind of distributed database.

Formal methods

- **FM** encompasses a **wide range** of techniques...
- Model checking:
  - constructing formal ***models***, with
  - appropriate formal ***specifications***.
- Example is **Promela** and **Spin**.

Promela and spin

- The **language** Promela is 'C' like, with an initialization procedure. It can model asynchronous or synchronous, deterministic or non-deterministic systems
- Spin is the **checker** for Promela models
- Assertions to test correctness of model:

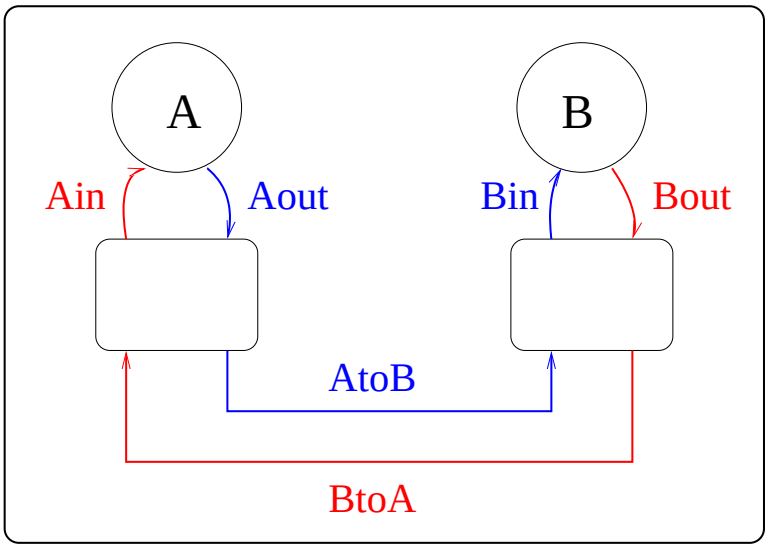
```
assert(some_boolean_condition);
```

- If condition not TRUE then assertion violated.

Temporal claims

- ***We got here again without making any progress!***
- The support for temporal claims takes the form of:
  - **Endstate** labels - for determining valid endstates
  - **Progress** labels - claim no non-progress cycles
  - **Never** claims - impossible temporal assertions

Simple example



Promela example

```
init
{
    chan AtoB = [1] of { mtype,byte };
    chan BtoA = [1] of { mtype,byte };
    chan Ain  = [2] of { mtype,byte };
    chan Bin  = [2] of { mtype,byte };
    chan Aout = [2] of { mtype,byte };
    chan Bout = [2] of { mtype,byte };
    atomic {
       run application( Ain,Aout );
       run transfer( Aout,Ain,BtoA,AtoB );
       run transfer( Bout,Bin,AtoB,BtoA );
       run application( Bin,Bout )
    };
    AtoB!err(0)
}
```

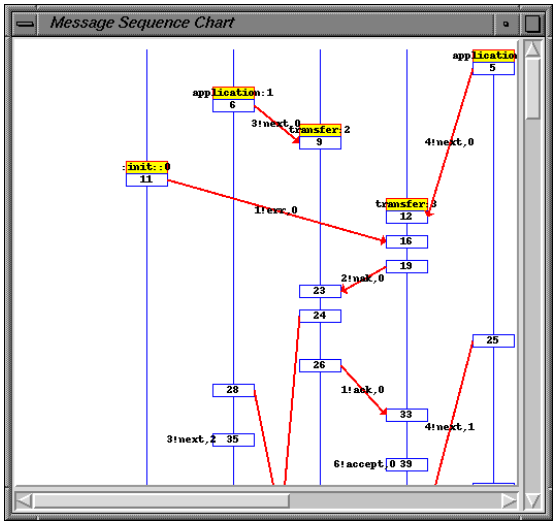
Promela example

```
#define MAX 10
mtype = { ack, nak, err, next, accept }
proctype transfer( chan in, out, chin, chout )
{
    byte o,i;
    in?next(o);
    do
      :: chin?nak(i) -> out!accept(i); chout!ack(o)
      :: chin?ack(i) -> out!accept(i); in?next(o); chout!ack(o)
      :: chin?err(i) -> chout!nak(o)
    od
}
```


Promela example

```
proctype application( chan in, out )
{
    int i=0, j=0, last_i=0;
    do
      :: in?accept(i) ->
            assert( i==last_i );
            if
              :: (last_i!=MAX) -> last_i = last_i+1
              :: (last_i==MAX)
            fi
      :: out!next(j) ->
            if
              :: (j!=MAX) -> j=j+1
              :: (j==MAX)
            fi
    od
}
```

Spin simulation



Formal evaluation - TCSEC

TCSEC (The Orange book) was the first rating system for the security of products. It defined six different evaluation classes. The classes are:

- **C1** - For **same-level** security access. Not currently used.
- **C2** - **Controlled access protection** - users are individually accountable for their actions. Most OS manufacturers have C2 versions of the OS.
- **B1** - **Mandatory BLP policies** - for more secure systems handling classified data.

Formal evaluation - TCSEC

- **B2** - **structured protection** - mandatory access control for all objects in the system. Formal models.
- **B3** - **security domains** - more controls, minimal complexity, provable consistency of model.
- **A1** - **Verified design** - consistency proofs between model and specification.

Formal evaluation - ITSEC

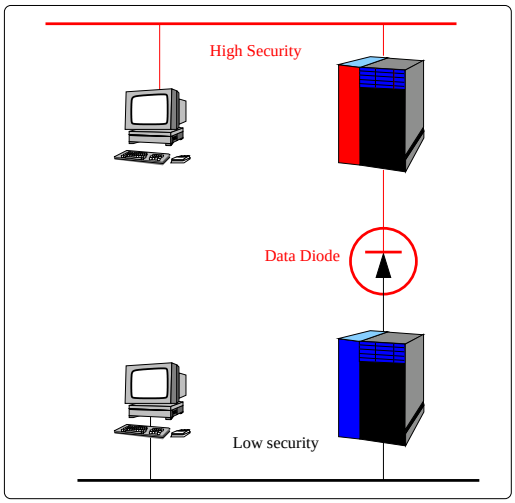
- From Dutch, English, French and German **national** security evaluation **criteria**.
- **Adaptable**.
- Sponsor determines operational requirements, threats and security objectives.
- ITSEC specifies the **interactions** and documents between the sponsor and the evaluator.

ITSEC

- Again there are various levels of evaluation: **E0..E6**, with **E6** giving the highest level of assurance - it requires two independant formal verifications.
- [Woo98] First **E6** certification of a smart-card system.
  - The smart-cards are electronic purses - that is they carry value,
  - Forgery must be impossible.
  - The certification encompassed the communication with the card, as well as the software within the card, and at the bank.

Data Diode E6, BLP

http://www.tenix.com/Main.asp?ID=908



Biometrics

Biometrics is the use of **human** physical **characteristics** to support **authentication**.

FPC1010 Area Sensor

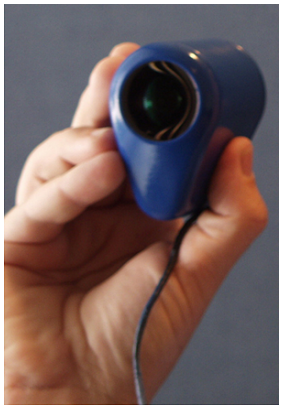
FEATURES

- Internal A/D
- SPI interface
- 3.3 Volt operation
- Robust surface coating
- >1 000 000 wear cycles
- >15kV ESD protection

APPLICATION EXAMPLES

- Mobile phones, PDAs
- PC peripherals
- Security systems
- Smart cards

Biometrics - eyes



Minimal hardware biometrics

- **Voices** - Record and process voice leading to either speaker verification or recognition.
- **Faces** - Capture either a static or moving image of a face.
- **Keystrokes** - capture a sequence of keystrokes, recording timing.

Combinations of characteristics may be used, but in general biometric techniques are **not reliable** on their own. Good second key for ***separation of privilege***.

Sample systems: PGP

- PGP (Pretty Good Privacy) is a public key encryption package to protect E-mail and data files.
- It lets you communicate securely with people you've never met, with no secure channels needed for prior exchange of keys.
- PGP can be used to append digital signatures to messages, as well as encrypt the messages, or do both.

PGP

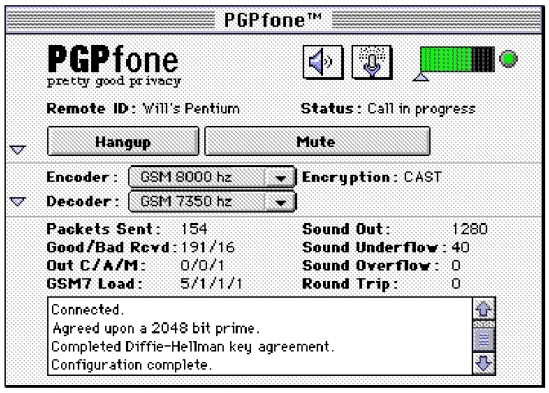
- It uses various schemes including patented ones like IDEA and RSA.
- The patent on IDEA allows non-commercial distribution, and the RSA patent has expired.
- However there are also commercial versions of PGP.
- PGP can use, for example, 2048 bit primes, and it is considered unlikely that PGP with this level of encryption can be broken.

PGPfone

- Speech compression and *strong cryptography*
- In 2002, it was available in two versions:
  1. An international version available *outside* America, and a prohibited import *into* America.
  2. An American version available *inside* America, and a prohibited import *out of* America.

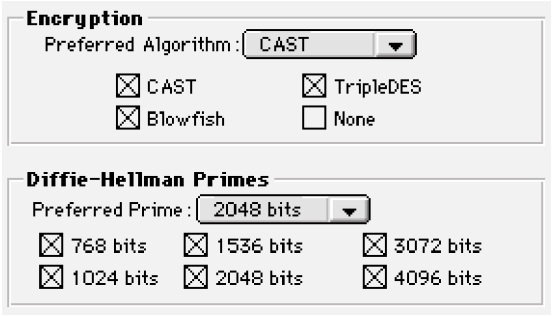
These two versions are also exactly the same! (Restrictions on the import and export of munitions - strong cryptography is considered a munition).

PGPfone



PGPfone

Familiar encryption and key exchange parameters:



When initially setting up a link, Diffie-Hellman key exchange is used to ensure safety in the choice of an encryption key.

Sample attack: CRC-32 on ssh

http://www.cert.org/incident_notes/IN-2001-12.html

Used in the matrix....

Sample attack: PkZip stream cipher

- PkZip is for compressing files
- PkZip can also scramble files when given a secret password.
- Enciphering strategy is weak and can be cracked
  - http://citeseer.ist.psu.edu/biham94known.html
- Weakness in the (homegrown) ciphering algorithm

PkZip stream cipher

```
opo 144% pkcrack -C all.zip -c readme.doc -P plain.zip -p readme.doc
Files read. Starting stage 1 on Wed Sep 8 09:04:02 1999
Generating 1st generation of possible key2_421 values...done.
Found 4194304 possible key2-values.
Now we're trying to reduce these...
Done. Left with 18637 possible Values. bestOffset is 24.
Stage 1 completed. Starting stage 2 on Thu Sep 9 09:12:06 1999
Ta-daaaaa! key0=dda9e469, key1=96212999, key2=f9fc9651
Probabilistic test succeeded for 402 bytes.
Stage2 completed. Starting password search on Thu Sep 9 09:22:22 1999
Key: 73 65 63 72 65 74
Or as a string: 'secret' (without the enclosing single quotes)
Finished on Thu Sep 9 10:54:22 1999 opo 99%
opo 145% ./zipdecrypt dda9e469 96212999 f9fc9651 all.zip rr.zip
opo 146%
```

rr.zip contains unencypted version of archive

PkZip stream cipher fix

The PkZip stream cipher is also susceptible to dictionary attacks, and so it is considered not suitable for secure encryption of data. The fix is:

Don't use PkZip for security purposes.

Someone wanted... voting protocols

Example with Alice, Bob and Charles (!), who vote and then encrypt and sign a series of messages using public-key encryption. For example, if Alice votes v_A, then she will broadcast to all other voters the message

$$R_A(R_B(R_C(E_A(E_B(E_C(v_A))))))$$

where R_A is a random encoding function which adds a random string to a message before encrypting it with A's public key, and E_A is public key encryption with A's public key.

Voting protocols

- Each voter then signs the message and decrypts one level of the encryption.
- At the end of the protocol, each voter has a complete signed audit trail and is ensured of the validity of the vote.

First round...

Who	Receives and removes one level...		and sends on...
Alice:	$R^1_A(R^1_B(R^1_C(E^1_A(E^1_B(E^1_C(v_1))))))$	$\rightarrow$	$R^1_B(R^1_C(E^1_A(E^1_B(E^1_C(v_1)))))$
	$R^2_A(R^2_B(R^2_C(E^2_A(E^2_B(E^2_C(v_2))))))$	$\rightarrow$	$R^2_B(R^2_C(E^2_A(E^2_B(E^2_C(v_2)))))$
	$R^3_A(R^3_B(R^3_C(E^3_A(E^3_B(E^3_C(v_3))))))$	$\rightarrow$	$R^3_B(R^3_C(E^3_A(E^3_B(E^3_C(v_3)))))$
Bob:	$R^1_B(R^1_C(E^1_A(E^1_B(E^1_C(v_1)))))$	$\rightarrow$	$R^1_C(E^1_A(E^1_B(E^1_C(v_1))))$
	$R^2_B(R^2_C(E^2_A(E^2_B(E^2_C(v_2)))))$	$\rightarrow$	$R^2_C(E^2_A(E^2_B(E^2_C(v_2))))$
	$R^3_B(R^3_C(E^3_A(E^3_B(E^3_C(v_3)))))$	$\rightarrow$	$R^3_C(E^3_A(E^3_B(E^3_C(v_3))))$
Charles:	$R^1_C(E^1_A(E^1_B(E^1_C(v_1))))$	$\rightarrow$	$E^1_A(E^1_B(E^1_C(v_1)))$
	$R^2_C(E^2_A(E^2_B(E^2_C(v_2))))$	$\rightarrow$	$E^2_A(E^2_B(E^2_C(v_2)))$
	$R^3_C(E^3_A(E^3_B(E^3_C(v_3))))$	$\rightarrow$	$E^3_A(E^3_B(E^3_C(v_3)))$

Voting protocols

- In the first round (after 3 transfers) - each voter has agreed that their vote has been counted.
- If not they do not continue and protocol finishes.
- To signify agreement, each person appends digital signatures to the votes they forward...

second round...

Who	Receives and removes one level...		and sends on...
Alice:	$E_A^1(E_B^1(E_C^1(v_1)))$	$\rightarrow$	$E_B^1(E_C^1(v_1))$
	$E_A^2(E_B^2(E_C^2(v_2)))$	$\rightarrow$	$E_B^2(E_C^2(v_2))$
	$E_A^3(E_B^3(E_C^3(v_3)))$	$\rightarrow$	$E_B^3(E_C^3(v_3))$
Bob:	$E_B^1(E_C^1(v_1))$	$\rightarrow$	$E_C^1(v_1)$
	$E_B^2(E_C^2(v_2))$	$\rightarrow$	$E_C^2(v_2)$
	$E_B^3(E_C^3(v_3))$	$\rightarrow$	$E_C^3(v_3)$
Charles:	$E_C^1(v_1)$	$\rightarrow$	v_1
	$E_C^2(v_2)$	$\rightarrow$	v_2
	$E_C^3(v_3)$	$\rightarrow$	v_3

Discussion

- Only Alice can remove her level of encryption, but anyone can check that it was done correctly (by re-encrypting).
- In the second round, someone can tamper with the vote, but...
  - at the end each vote can be re-encrypted, and checked against the set of signatures..
  - tamperer will be found.
- Unwieldy

Revision: topics

- ✔ Math preliminaries (xor, modulo, primes, GF, entropy)
- ✔ Mono and poly alphabetic ciphers.
- ✔ BLP and Biba models.
- ✔ RSA ... work through math

A look at the exam paper...

Exam

- 21/11/2005 - Monday evening at 5:00 in S16 SR5, S16 TW4
- You can expect **13 pages** -
  - write on paper.
  - Marks/50.
  - Open book.
  - No computers.

Exam coverage

- **10 short answer questions** worth 1 mark each
- Longer questions on...
  - **Encryption (15 marks)**
  - **Checksums/Signatures (4 marks)**
  - **Preliminaries (10 marks)**
  - **Models (5 marks)**
  - **Protocols (6 marks)**